Worksheet 11
Camera Distance Calibration

Learning Outcome 1,4

Book Chapter 5

Purpose:

There is a relationship between object **width** returned by the Pixy API and the distance of the object from the camera. If we know this, then we can deduce the object distance from the measurement of **width.**

We know from the theory in Chapter 5, that the distance to an object is proportional to the inverse of the object width. This is expressed as the straight line

$$dist = gradient\left(\frac{1}{width}\right) + intercept$$

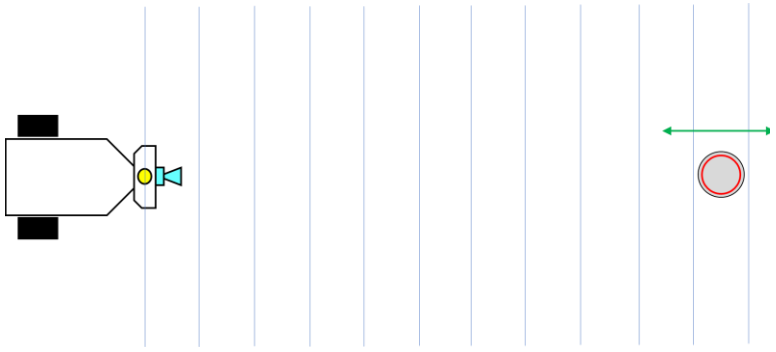
where two parameters, the *gradient* and the *intercept* of the line need to be found.

We perform an experiment to get *distance-width* data, then fit a straight line approximately by hand. Then we call a non-linear regression script to find the optimal values of *gradient* and *intercept* from our data. This enables us to create a function like this, in code.

```
float getDistanceFromObject(uint16_t width){
  float dist;
  dist = gradient/(float)width + intercept;
  return dist;
}
```

1. Collecting Data and Manual line fitting

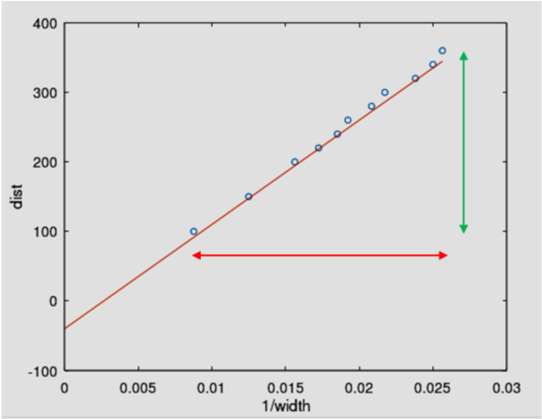
(a) Find a suitably large flat area (such as the arena) for the robot to move on. Then create a grid of lines of suitable spacing (you can choose) on some sheets paper sellotaped together. Here's a plan diagram of what to do. The robot is shown, and the red object which you will move. The largest distance should be the largest distance you will use the camera in future activities. ***I suggest at least 400 mm.***



Object-Robot Distance (mm)	Reported Image Width (pels)

(b) Run the sketch **CBP_2403_R_Dist_Measure** which will report the width of the image of the red object. Move the object at on your grid and record a table of physical distances (mm) and image widths (pixels). Make up a table like the one on the left.

(c) Download the folder **Octave Assets** to the desktop and fire up Octave and point it to the folder.



(d) Open the Octave file **distance_LUT_fit.m** and over-write data in matrix **A = […];** with yours. Run the script (command line). You will see your data and a line which has no meaning at all. Here's what I got, on the left, (after I fitted the red line, your red line will be really off at first). If your data does not get plotted over the whole graph, comment out (using **%**) the following line in the Octave script **%line(x,y);**

(e) We have to find the *gradient* of this line. Estimate the dist range in your data corresponding to the green arrow (I get 350 – 100 = 250) and the 1/width range (I get 0.026 – 0.01 = 0.016). The gradient estimate is 250/0.016 = 15625.

(f) Now find the intercept, the dist value where 1/width = 0. I get around -30.

(g) Now stuff these values into the Octave script where indicated.

2. Automated line fitting

Here we shall use an algorithm built-in to Octave to find the line which fits our data optimally. We shall use your estimated slope and intercept parameters to make an automated fit.

(a) Open the script **PixyDist.m** and copy and paste your data into matrix **A = […];** over-writing my values.

(b) Put your estimated *gradient* and *intercept* into the script (near line 33) replacing my values.

(c) Run the script and look at the nice optimal fit of distance to width. Look at the **r2** value, I get 0.997. This means that the curve explains 99.7% of the data, very acceptable.

(b) Note down the coefficients (values of parameters) reported. I get 15276.2 for the gradient and -38.3 for the intercept. This means that I can write the following function to get distance from width. See where I put in the coefficients.

```
float getDistanceFromObject(uint16_t width)
{
  float dist
  dist = (15276.2)/(float)width – 38.3;
  return dist;
}
```

4. Testing your Calibration

(a) Open up the sketch **CBP_2403_R_Dist_Test** and look for the above function. Replace my parameters with yours.

(b) Run the sketch and fire up the Serial Monitor. Place the red object in PixyCam's field of view *at a few measured distances*. Compare the physical distance with the distance recorded, according to your calibration. If it doesn't work, I have some tissues to mop-up your tears.

All done!